
MMM MMM AAAAAAAAA CCCCCCCCCCC RRRRRRRRRRR 000000000
MMM MMM AAAAAAAAA CCCCCCCCCCC RRRRRRRRRRR 000000000
MMM MMM AAAAAAAAA CCCCCCCCCCC RRRRRRRRRRR 000000000
MMMMMM MMMMMM AAA AAA CCC RRR RRR OOO OOO
MMMMMM MMMMMM AAA AAA CCC RRR RRR OOO OOO
MMMMMM MMMMMM AAA AAA CCC RRR RRR OOO OOO
MMM MMM MMM AAA AAA CCC RRR RRR OOO OOO
MMM MMM MMM AAA AAA CCC RRR RRR OOO OOO
MMM MMM MMM AAA AAA CCC RRR RRR OOO OOO
MMM MMM AAA AAA CCC RRRRRRRRRRR OOO OOO
MMM MMM AAA AAA CCC RRRRRRRRRRR OOO OOO
MMM MMM AAA AAA CCC RRRRRRRRRRR OOO OOO
MMM MMM AAAAAAAAAAAAAAA CCC RRR RRR OOO OOO
MMM MMM AAAAAAAAAAAAAAA CCC RRR RRR OOO OOO
MMM MMM AAAAAAAAAAAAAAA CCC RRR RRR OOO OOO
MMM MMM AAA AAA CCC RRR RRR OOO OOO
MMM MMM AAA AAA CCC RRR RRR OOO OOO
MMM MMM AAA AAA CCC RRR RRR OOO OOO
MMM MMM AAA AAA CCCCCCCCCCC RRR RRR 000000000
MMM MMM AAA AAA CCCCCCCCCCC RRR RRR 000000000
MMM MMM AAA AAA CCCCCCCCCCC RRR RRR 000000000
MMM MMM AAA AAA CCCCCCCCCCC RRR RRR 000000000

FILEID**BDYSCN

```
BBBBBBBB    DDDDDDDD    YY      YY    SSSSSSSS    CCCCCCCC  NN      NN
BBBBBBBB    DDDDDDDD    YY      YY    SSSSSSSS    CCCCCCCC  NN      NN
BB      BB  DD      DD  YY      YY  SS          CC          NN      NN
BB      BB  DD      DD  YY      YY  SS          CC          NN      NN
BB      BB  DD      DD    YY  YY    SS          CC          NNNN    NN
BB      BB  DD      DD    YY  YY    SS          CC          NNNN    NN
BBBBPP9B    DD      DD      YY        SSSSSS    CC          NN  NN  NN
BBBBBBBB    DD      DD      YY        SSSSSS    CC          NN  NN  NN
BB      BB  DD      DD      YY              SS  CC          NN    NNNN
BB      BB  DD      DD      YY              SS  CC          NN    NNNN
BB      BB  DD      DD      YY              SS  CC          NN      NN      ....
BB      BB  DD      DD      YY              SS  CC          NN      NN      ....
BBBBBBBB    DDDDDDDD        YY      SSSSSSSS      CCCCCCCC  NN      NN      ....
BBBBBBBB    DDDDDDDD        YY      SSSSSSSS      CCCCCCCC  NN      NN      ....

LL              IIIIII      SSSSSSSS
LL              IIIIII      SSSSSSSS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II        SSSSSS
LL                II        SSSSSS
LL                II              SS
LL                II              SS
LL                II              SS
LL                II              SS
LLLLLLLLLL      IIIIII    SSSSSSSS
LLLLLLLLLL      IIIIII    SSSSSSSS
```

```
0000     1          .TITLE  MAC$BDYSCN      SCAN MACRO BODY
0000     2          .IDENT  'V04-000'
0000     3
0000     4  ;
0000     5  ;**********************************************************************************
0000     6  ;*                                                                                *
0000     7  ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                       *
0000     8  ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                        *
0000     9  ;*  ALL RIGHTS RESERVED.                                                          *
0000    10  ;*                                                                                *
0000    11  ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED          *
0000    12  ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE        *
0000    13  ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER        *
0000    14  ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY        *
0000    15  ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY        *
0000    16  ;*  TRANSFERRED.                                                                  *
0000    17  ;*                                                                                *
0000    18  ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE         *
0000    19  ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT        *
0000    20  ;*  CORPORATION.                                                                  *
0000    21  ;*                                                                                *
0000    22  ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS        *
0000    23  ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                       *
0000    24  ;*                                                                                *
0000    25  ;*                                                                                *
0000    26  ;**********************************************************************************
0000    27  ;
0000    28
0000    29  ;++
0000    30  ; FACILITY:     VAX MACRO ASSEMBLER OBJECT LIBRARY
0000    31  ;
0000    32  ; ABSTRACT:
0000    33  ;
0000    34  ; The VAX-11 MACRO assembler translates MACRO-32 source code into object
0000    35  ; modules for input to the VAX-11 LINKER.
0000    36  ;
0000    37  ; ENVIRONMENT:  USER MODE
0000    38  ;
0000    39  ; AUTHOR: Benn Schreiber, CREATION DATE: 20-AUG-78
0000    40  ;
0000    41  ; MODIFIED BY:
0000    42  ;
0000    43  ;       V01.14  RN0029          R. Newland       9-Feb-1980
0000    44  ;               Copy macro comments correctly into virtual memory.
0000    45  ;               SPR 11-28597
0000    46  ;
0000    47  ;       V01.13  RN0023          R. Newland       3-Nov-1979
0000    48  ;               New message codes to get error messages from system
0000    49  ;               message file.
0000    50  ;
0000    51  ;       V01.12  RN0010          R. Newland       5-Sep-1979
0000    52  ;               Multipage MXB blocks
0000    53  ;
0000    54  ;       V01.14  RN0028          R. Newland      18-Jan-1980
0000    55  ;               Do not process lexical functions for macros defined
0000    56  ;               within another macro.  SPR 11-28098
0000    57  ;
```

```
0000     58 ;
0000     59 ;       V01.14  RN0025          R. Newland      15-Dec-1979
0000     60 ;               Copy symbols correctly from macro body.
0000     61 ;               SPR 11-27450
0000     62 ;
0000     63 ;       V01.11  RN0005          R. Newland      13-Aug-1979
0000     64 ;               Remove .ALIGN LONG and .DEBUG statements
0000     65 ;--
```

```
                     0000     67          .SBTTL  DECLARATIONS
                     0000     68 ;
                     0000     69 ; INCLUDE FILES:
                     0000     70 ;
                     0000     71
                     0000     72 ;
                     0000     73 ; MACROS:
                     0000     74 ;
                     0000     75          $MAC_CTLFLGDEF                  ;DEFINE CONTROL FLAGS
                     0000     76          $MAC_GENVALDEF                  ;DEFINE GENERAL VALUES
                     0000     77          $MAC_INTCODDEF                  ;DEFINE INTER. BUFFER CODES
                     0000     78          $MAC_MTXDEF                     ;DEFINE SPECIAL MACRO TEXT OPERATOR BYTES
                     0000     79          $MAC_SYMBLKDEF                  ;DEFINE SYMBOL BLOCK OFFSETS
                     0000     80          $MAC_MNBDEF                     ;DEFINE MNB OFFSETS
                     0008     81          $MACMSGDEF                      ; Define message codes
                     0008     82
                     0008     83 ;
                     0008     84 ; EQUATED SYMBOLS:
                     0008     85 ;
                     0008     86
                     0008     87 ;
                     0008     88 ; LOCAL DATA
                     0008     89 ;
                00000000     90          .PSECT  MAC$RO_DATA,NOEXE,NOWRT,GBL,LONG
                     0000     91
           00000000  0000     92          INSYMP  =       0
                     0000     93
                     0000     94          $MAC_INSERT_SYM EXTRACT,LEXICAL_EXTRACT,MTX$_LXEXT ;%EXTRACT
                     0015     95          $MAC_INSERT_SYM LENGTH,LEXICAL_LENGTH,MTX$_LXLEN   ;%LENGTH
                     0029     96          $MAC_INSERT_SYM LOCATE,LEXICAL_LOCATE,MTX$_LXLOC,- ;%LOCATE
                     0029     97                          MAC$LEX_OP_LIST
                     003D     98
                00000000     99          .PSECT  MAC$RO_CODE_MAC,NOWRT,GBL,LONG
```

```
                     0000   101          .SBTTL  MAC$BODY_SCAN   SCAN MACRO BODY
                     0000   102
                     0000   103 ;++
                     0000   104 ; FUNCTIONAL DESCRIPTION:
                     0000   105 ;
                     0000   106 ;       THIS ROUTINE IS CALLED TO SCAN A MACRO, REPEAT OR INDEFINITE
                     0000   107 ;       REPEAT BODY AND READ IT INTO VIRTUAL MEMORY.
                     0000   108 ;
                     0000   109 ; CALLING SEQUENCE:
                     0000   110 ;
                     0000   111 ;       CALLX   X,MAC$BODY_SCAN
                     0000   112 ;
                     0000   113 ; INPUTS:
                     0000   114 ;
                     0000   115 ;       4(AP)           POINTER TO WHERE TO START STORING TEXT
                     0000   116 ;       8(AP)           NO. OF BYTES LEFT IN BLOCK POINTED TO BY 4(AP)
                     0000   117 ;       12(AP)          POINTER TO MACRO ARGUMENT LIST OR 0 IF NO ARGS
                     0000   118 ;
                     0000   119 ; OUTPUTS:
                     0000   120 ;
                     0000   121 ;       R0              0 IF NULL MACRO BODY, ELSE NON-ZERO
                     0000   122 ;
                     0000   123 ;       THE MACRO TEXT WILL BE STORED AT THE APPOINTED LOCATION.  ANY
                     0000   124 ;       ADDITIONAL PAGES REQUIRED WILL BE GOTTEN AND THE PAGES WILL BE
                     0000   125 ;       LINKED INTO THE MACRO NAME BLOCK PAGE LIST (MNB$L_PAGP).  UNLESS
                     0000   126 ;       FLG$M_RPTIRP IS SET THE ADDITIONAL PAGES WILL BE COUNTED IN THE
                     0000   127 ;       NUMBER OF PAGES USED TO DEFINE MACROS.
                     0000   128 ;
                     0000   129 ;--
                     0000   130
                     0000   131 MAC$BODY_SCAN::
                     0000   132          .WORD   ^M<R6,R7,R8>            ;REGISTER SAVE MASK
                  7E 01C0   0000
                     0002   133          CLRL    -(SP)                   ;ASSUME NULL MACRO BODY
                  CF 01 D0  0004   134          MOVL    #1,W^MAC$GL_MCLVL       ;START AT MACRO LEVEL 1
                  AC 04 D0  0009   135          MOVL    4(AP),R8                ;POINT TO THE BLOCK
               57 AC 06 C3  000D   136          SUBL3   #6,8(AP),R7             ;GET CHARACTER COUNT AND LEAVE
                     0012   137                                                  ;ROOM FOR LINK FLAG WORD AND LINK
                  0A 14     0012   138          BGTR    20$                     ;BRANCH IF THERE IS ROOM
               88 FE 8F 99  0014   139          CVTBW   #MTX$_TXTLNK,(R8)+      ;NO--START WITH A LINK
                  FFE5' 30  0018   140          BSBW    MAC$ALL_1_PAGE          ;ALLOCATE A PAGE
                  0343 30   001B   141          BSBW    MAC$VSTO_GET_0          ;SETUP LINK
                            001E   142 20$:
```

```
                     001E   144 ;
                     001E   145 ; COME HERE AT THE BEGINNING OF EACH NEW LINE.  THE ADDRESS OF THE COUNT
                     001E   146 ; WORD FOR THIS LINE IS STACKED SO WE CAN STORE THE LINE COUNT THERE AT
                     001E   147 ; THE END OF THE LINE.  FIRST WE MUST CHECK TO SEE IF THERE IS ROOM TO
                     001E   148 ; STORE THE COUNT WORD.  IF NOT, GET A NEW PAGE AND SET THE SPECIAL
                     001E   149 ; COUNT WORD LINK FLAG.
                     001E   150 ;
                     001E   151
                     001E   152 BODY_SCAN_LOOP:
      00 6B    22    E5  001E   153          BBCC    #FLG$V_SPECOP,(R11),.+1 ;CLEAR SPECIAL OPERATOR FLAG
               58    DD  0022   154          PUSHL   R8                      ;SAVE PTR TO BEGINNING OF LINE
   50  57    08    C3  0024   155          SUBL3   #8,R7,R0                ;SEE IF ROOM FOR COUNT, TXTLNK, AND LINK
               0D    14  0028   156          BGTR    10$                     ;IF GTR YES
      88   FE 8F    99  002A   157          CVTBW   #MTX$_TXTLNK,(R8)+      ;NO--START WITH A LINK
           FFCF'    30  002E   158          BSBW    MAC$ALL_1_PAGE          ;ALLOCATE A PAGE
           032D     30  0031   159          BSBW    MAC$VSTO_GET_0          ;SETUP LINK
         6E    58    D0  0034   160          MOVL    R8,(SP)                 ;UPDATE BEGINNING OF LINE PTR
               88    B4  0037   161 10$:     CLRW    (R8)+                   ;CLEAR BYTE COUNT (FILLED IN LATER)
               56    D4  0039   162          CLRL    R6                      ;INIT COUNT OF CHARS STORED
         57    02    C2  003B   163          SUBL2   #2,R7                   ;DECREMENT BYTES REMAINING
   5A    0000'CF    9A  003E   164          MOVZBL  W^MAC$AB_LINEBF,R10     ;COPY 1ST CHAR IN CASE ";"
6B  04000009 8F    CA  0043   165          BICL2   #FLG$M_ALLCHR!FLG$M_CONT!FLG$M_SPLALL,(R11) ;DO NOT PASS ALL CHARS
                     004A   166                                              ;AND NO CONT. LINES
      0E 0005'CF    E8  004A   167          BLBS    W^LST$G_MACRODEF+SYM$L_VAL,20$ ;BRANCH IF LISTING MACRO DEFS
                     004F   168          $INTOUT_LW INT$_SETLONG,<#0,#MAC$GL_LIST_IT> ;NO--SEND TO PASS 2
      00 6B    05    E3  005D   169 20$:     BBCS    #FLG$V_ENDMCH,(R11),30$ ;ALLOW DIRECTIVE CHECKING
                     0061   170 ;
                     0061   171 ; HERE FOR EACH NEW CHARACTER IN THE LINE
                     0061   172 ;
         25    5A    91  0061   173 30$:     CMPB    R10,#^A/%/              ;POSSIBLY A LEXICAL OPERATOR?
               0C    12  0064   174          BNEQ    35$                     ;IF NEQ NO
   01    0000'CF    D1  0066   175          CMPL    W^MAC$GL_MCLVL,#1       ; In outer macro (the macro being defined)?
               05    12  006B   176          BNEQ    35$                     ; No if NEQ, do not process lexical fns
            00E6    30  006D   177          BSBW    SCAN_PERCENT            ;YES--CHECK IT OUT
              EF    11  0070   178          BRB     30$                     ;CONTINUE SCANNING WITH NEXT
                     0072   179                                              ;CHARACTER AFTER THE OPERATOR
         0D    5A    91  0072   180 35$:     CMPB    R10,#CR                 ;END OF LINE?
               3C    13  0075   181          BEQL    40$                     ;IF EQL YES--DO END OF LINE PROCESSING
               02    E1  0077   182          BBC     #CHR$V_SYM_CHR,-        ;BRANCH IF CHAR CANNOT
      39 0000'CA         0079   183                  W^MAC$AB_CHSK_TAB(R10),42$ ;START A SYMBOL
         0000'CF    DD  007D   184          PUSHL   W^MAC$GL_LINEPT         ; Save line pointer
         6B    01    C8  0081   185          BISL2   #FLG$M_ALLCHR,(R11)     ; Pass all characters
            FF79'    30  0084   186          BSBW    MAC$SYMSCNUP1           ; Scan a symbol
         6B    01    CA  0087   187          BICL2   #FLG$M_ALLCHR,(R11)     ; Clear ALLCHR flag
               54 8ED0  008A   188          POPL    R4                      ; Reset stack and get line pointer
         3A    5A    91  008D   189          CMPB    R10,#^A/:/              ;IS IT A LABEL?
               26    13  0090   190          BEQL    50$                     ;IF EQL YES--DON'T CHECK IT
      26 6B    05    E5  0092   191          BBCC    #FLG$V_ENDMCH,(R11),60$ ;NO--CHECKING DIRECTIVES?
                     0096   192                                              ;(AND DISABLE CHECKING)
   2E    0001'CF    91  0096   193          CMPB    W^MAC$AB_TMPSYM+1,#^A/./;YES--CAN SYMBOL BE A DIRECTIVE?
               1F    12  009B   194          BNEQ    60$                     ;IF NEQ NO
55  00000000'EF    9E  009D   195          MOVAB   MAC$G_SPLMC_DIR,R5      ;POINT TO LIST
           FF59'    30  00A4   196          BSBW    MAC$SRC_LIST            ;SEE IF SPECIAL MACRO DIRECTIVE
            12 50   E9  00A7   197          BLBC    R0,60$                  ;BRANCH IF NOT FOUND
            05 B1   16  00AA   198          JSB     @SYM$L_VAL(R1)          ;YES--PROCESS THE DIRECTIVE
         0000'CF    D5  00AD   199          TSTL    W^MAC$GL_MCLVL          ;STILL IN A MACRO?
               43    14  00B1   200          BGTR    90$                     ;IF GTR YES
```

```
                     0066   31  00B3   201 40$:    BRW     BODY_SCAN_END           ;NO--EXIT
                       57   11  00B6   202 42$:    BRB     100$                    ;BRANCH AID
                00 6B  05   E3  00B8   203 50$:    BBCS    #FLG$V_ENDMCH,(R11),60$ ;JUST SCANNED A LABEL, ALLOW
                                00BC   204                                          ;CHECKING OF DIRECTIVES
                                00BC   205 60$:
                                00BC   206 ;
                                00BC   207 ; SEE IF A MACRO ARGUMENT
                                00BC   208 ;
                55   0C AC   D0  00BC   209         MOVL    12(AP),R5               ;POINT TO MACRO ARG NAMES
                         3E   13  00C0   210         BEQL    95$                     ; If EQL no macro argument names
                       FF3B'  30  00C2   211 65$:    BSBW    MAC$SRC_LIST            ;SEE IF A MACRO ARG
                     38 50    E9  00C5   212         BLBC    R0,95$                  ; Branch if not
                27   FF A8   91  00C8   213         CMPB    -1(R8),#^A/'/           ;YES--CONCATENATION PRECEDING
                                00CC   214                                          ;MACRO ARG?
                         06   12  00CC   215         BNEQ    70$                     ;IF NEQ NO
                         58   D7  00CE   216         DECL    R8                      ;YES--BACKUP POINTER
                         57   D6  00D0   217         INCL    R7                      ;ADJUST COUNTER
                         56   D7  00D2   218         DECL    R6                      ;DECREMENT COUNT OF CHARS STORED
                   50    02   D0  00D4   219 70$:    MOVL    #2,R0                   ;SET TO STORE TWO BYTES
                         51   DD  00D7   220         PUSHL   R1                      ;SAVE MAB ADDRESS
                       0269   30  00D9   221         BSBW    MAC$VSTO_SET            ;...
                88     FF 8F   90  00DC   222         MOVB    #MTX$_ARGMRK,(R8)+      ;MARK MACRO ARG
                00 6B    22    E2  00E0   223         BBSS    #FLG$V_SPECOP,(R11),.+1 ;FLAG SPECIAL OPERATOR ON LINE
                         51 8ED0  00E4   224         POPL    R1                      ;RESTORE MAB ADDRESS
                88     05 A1   90  00E7   225         MOVB    MAB$B_ARGNO(R1),(R8)+   ;STORE ARGUMENT NUMBER
                   27    5A   91  00EB   226         CMPB    R10,#^A/'/              ;POST-CONCATENATION?
                         03   12  00EE   227         BNEQ    80$                     ;IF NEQ NO
                       FF0D'  30  00F0   228         BSBW    MAC$GETCHR              ;YES--EAT THE QUOTE
                       FF6B   31  00F3   229 80$:    BRW     30$                     ;PROCESS NEXT CHARACTER
                                00F6   230 ;
                                00F6   231 ; COPY DIRECTIVE TO OUTPUT
                                00F6   232 ;
                52   0000'CF   9E  00F6   233 90$:    MOVAB   W^MAC$AB_TMPSY1,R2      ;POINT TO DIRECTIVE NAME
                     53   82   9A  00FB   234         MOVZBL  (R2)+,R3                ;GET LENGTH OF NAME
                          0A   11  00FE   235         BRB     98$
                                0100   236 ;
                                0100   237 ; Copy non-argument string to output
                                0100   238 ;
                                0100   239 95$:
                52   54   01   C3  0100   240         SUBL3   #1,R4,R2                ; Get address of string
             53 0000'CF   54   C3  0104   241         SUBL3   R4,W^MAC$GL_LINEPT,R3   ; and compute its size
                                010A   242 98$:
                         0222   30  010A   243         BSBW    MAC$VSTO_STR            ;COPY TO VIRTUAL BUFFER
                           E4   11  010D   244         BRB     80$                     ;PROCESS NEXT CHARACTER
                                010F   245 ;
                                010F   246 ; CHARACTER NOT SYMBOL CONSTITUENT
                                010F   247 ;
                         020D   30  010F   248 100$:   BSBW    MAC$VSTO_CHAR           ;STORE THE CHARACTER
                     6B    01   C8  0112   249         BISL2   #FLG$M_ALLCHR,(R11)     ;PASS ALL CHARS (SEMICOLONS)
                         FEE8'  30  0115   250         BSBW    MAC$GETCHR              ;GET NEXT CHAR
                   D7 6B   00   E4  0118   251         BBSC    #FLG$V_ALLCHR,(R11),80$ ;CLEAR ALLCHR FLAG AND CONTINUE
                                011C   252                                          ;SCANNING.
                                011C   253
                                011C   254 BODY_SCAN_END:
                00 6B    1A   E5  011C   255         BBCC    #FLG$V_SPLALL,(R11),.+1 ;CLEAR SPECIAL ALL CHARS FLAG
                         51 8ED0  0120   256         POPL    R1                      ;GET ADDRESS TO STORE LINE LENGTH AT
                05 6B    22   E5  0123   257         BBCC    #FLG$V_SPECOP,(R11),10$ ;BRANCH IF NO SPECIAL OPERATORS ON LINE
```

```
                56   8000 8F    A8  0127   258          BISW2   #^X8000,R6              ;THERE WERE--FLAG IN LINE LENGTH
                61     56   B0  012C   259 10$:        MOVW    R6,(R1)                 ;STORE LINE LENGTH
                         02   13  012F   260          BEQL    20$                     ;BRANCH IF NULL LINE
                         6E   D6  0131   261          INCL    (SP)                    ;FLAG NOT NULL MACRO NOW
                0000'CF      D5  0133   262 20$:        TSTL    W^MAC$GL_MCLVL          ;STILL IN MACRO?
                         09   15  0137   263          BLEQ    30$                     ;IF LEQ NO
                5A       0D   9A  0139   264          MOVZBL  #CR,R10                 ;YES--FORCE READING NEW LINE
                    FEC1'    30  013C   265          BSBW    MAC$GETCHR              ;READ NEW CHAR/LINE
                    FEDC     31  013F   266          BRW     BODY_SCAN_LOOP          ;CONTINUE SCANNING MACRO BODY
                50       02   D0  0142   267 30$:        MOVL    #2,R0                   ;SET TO STORE 2 BYTES
                    01FD     30  0145   268          BSBW    MAC$VSTO_SET            ;...
                88   FFFF 8F    B0  0148   269          MOVW    #-1,(R8)+               ;MARK END WITH LINE LENGTH OF -1
                0000'CF      58  D0  014D   270          MOVL    R8,W^MAC$GL_ARGPTR      ;STORE UPDATED ARG POINTER
                         50 8ED0  0152   271          POPL    R0                      ;GET THE NULLNESS FLAG
                              04  0155   272          RET                             ;EXIT FROM BODY SCAN
```

```
                     0156   274 ;
                     0156   275 ; THIS ROUTINE IS CALLED WHEN A PERCENT SIGN IS ENCOUNTERED WHILE LOADING
                     0156   276 ; MACRO TEXT INTO CORE.  IT COMPLETELY PROCESSES THE LEXICAL OPERATOR
                     0156   277 ; INTO SPECIAL MTX$_ BYTES AND RETURNS WITH THE CHARACTER FOLLOWING
                     0156   278 ; THE LEXICAL OPERATOR.
                     0156   279 ;
                     0156   280
                     0156   281 SCAN_PERCENT:
    0000'CF    DD    0156   282         PUSHL   W^MAC$GL_LINEPT          ;SAVE CURRENT LINE POINTER
    6B   01    C8    015A   283         BISL2   #FLG$M_ALLCHR,(R11)      ;PASS ALL CHARACTERS BACK
       FEA0'   30    015D   284         BSBW    MAC$GETCHR               ;GET THE NEXT CHARACTER
    6B   01    CA    0160   285         BICL2   #FLG$M_ALLCHR,(R11)      ;CLEAR ALL CHARACTER FLAG
         02    E1    0163   286         BBC     #CHR$V_SYM_CHR,-         ;BRANCH IF CHARACTER IS NOT A SYMBOL
 23 0000'CA          0165   287                 W^MAC$TAB_CHSK_TAB(R10),10$ ;CONSTITUENT
                     0169   288 ;
                     0169   289 ; MAYBE WE HAVE A LEXICAL OPERATOR.  SCAN THE SYMBOL
                     0169   290 ;
       FE94'   30    0169   291         BSBW    MAC$SYMSCNUP1            ;ACCUMULATE THE SYMBOL NAME
    28   5A    91    016C   292         CMPB    R10,#^A/(/               ;SHOULD STOP ON LEFT PAREN
         1B    12    016F   293         BNEQ    10$                      ;IF NEQ NOT A LEXICAL OPERATOR
 55  0030'CF    9E    0171   294         MOVAB   W^MAC$LEX_OP_LIST,R5     ;LOAD UP OPERATOR NAME TABLE ADDRESS
       FE87'   30    0176   295         BSBW    MAC$SRC_LIST             ;SEE IF A LEXICAL OPERATOR
      10 50    E9    0179   296         BLBC    R0,10$                   ;IF LBC NO--FIX UP AND RETURN
         50 8ED0    017C   297         POPL    R0                       ;CLEAR SAVED LINE POINTER FROM STACK
      05 A1    DD    017F   298         PUSHL   SYM$L_VAL(R1)            ;YES--STACK PROCESSOR ADDRESS
 5A   09 A1    9A    0182   299         MOVZBL  SYM$W_FLAG(R1),R10       ;GET THE MTX IDENTIFIER BYTE
 0A 6B   22    E2    0186   300         BBSS    #FLG$V_SPECOP,(R11),20$  ;FLAG SPECIAL OPERATOR ON LINE AND
         08    11    018A   301         BRB     20$                      ;GO STORE THE MTX BYTE, GET THE
                     018C   302                                          ;NEXT CHARACTER, AND DISPATCH
                     018C   303                                          ;TO PROCESSOR
0000'CF 8ED0         018C   304 10$:     POPL    W^MAC$GL_LINEPT          ;RESTORE OLD LINE POINTER
5A   25    D0    0191   305         MOVL    #^A/%/,R10               ;RESTORE THE PERCENT SIGN
   0188    30    0194   306 20$:     BSBW    MAC$VSTO_CHAR            ;SEND IT TO THE OUTPUT
6B   01    C8    0197   307         BISL2   #FLG$M_ALLCHR,(R11)       ;PASS ALL CHARACTERS
   FE63'   30    019A   308         BSBW    MAC$GETCHR               ;GET NEXT CHARACTER
6B   01    CA    019D   309         BICL2   #FLG$M_ALLCHR,(R11)       ;CLEAR ALLCHR FLAG
         05    01A0   310         RSB                              ;RETURN
```

```
                     01A1   312          .SBTTL  PROCESS %EXTRACT LEXICAL OPERATOR
                     01A1   313
                     01A1   314 ;++
                     01A1   315 ; FUNCTIONAL DESCRIPTION:
                     01A1   316 ;
                     01A1   317 ;       THIS ROUTINE PROCESSES THE %EXTRACT LEXICAL OPERATOR.  THE
                     01A1   318 ;       ARGUMENTS TO THE OPERATOR ARE %EXTRACT(POS,LEN,STRING).
                     01A1   319 ;
                     01A1   320 ;--
                     01A1   321
                     01A1   322 LEXICAL_EXTRACT:
   0A    10          01A1   323          BSBB    10$                     ;GET THE POSITION AND SKIP COMMA
   08    10          01A3   324          BSBB    10$                     ;GET LENGTH AND SKIP THE COMMA
 00D6    30          01A5   325          BSBW    GET_LEX_STRING          ;GET THE STRING TO EXTRACT FROM
51 50    E9          01A8   326          BLBC    R0,LEXICAL_ARG_ERR      ;BRANCH IF ERROR
   69    11          01AB   327          BRB     LEXICAL_END_CK          ;FINISH UP
                     01AD   328 ;
                     01AD   329 ; GET A NUMBER AND SKIP THE COMMA AFTER IT--NO RETURN IF ERROR
                     01AD   330 ;
 008F    30          01AD   331 10$:     BSBW    GET_LEX_NUM             ;GET THE NUMBER
07 50    E9          01B0   332          BLBC    R0,20$                  ;BRANCH IF ERROR
  0130   30          01B3   333          BSBW    LEX_SKIP_COMMA          ;SKIP THE COMMA
01 50    E9          01B6   334          BLBC    R0,20$                  ;BRANCH IF ERROR
   05    05          01B9   335          RSB
   8E    D5          01BA   336 20$:     TSTL    (SP)+                   ;CLEAR CALL FROM STACK
   3E    11          01BC   337          BRB     LEXICAL_ARG_ERR         ;PROCESS THE ERROR
                     01BE   338
                     01BE   339 ;++
                     01BE   340 ; FUNCTIONAL DESCRIPTION:
                     01BE   341 ;
                     01BE   342 ;       THIS ROUTINE PROCESSES THE %LOCATE LEXICAL FUNCTION.  THE
                     01BE   343 ;       ARGUMENTS TO LOCATE ARE %LOCATE(SUBSTRING,STRING,STARTPOS),
                     01BE   344 ;       WITH THE STARTPOS ARGUMENT BEING OPTIONAL.
                     01BE   345 ;
                     01BE   346 ;--
                     01BE   347
                     01BE   348 LEXICAL_LOCATE:
  00BD    30         01BE   349          BSBW    GET_LEX_STRING          ;GET THE SUBSTRING TO LOCATE
 38 50    E9         01C1   350          BLBC    R0,LEXICAL_ARG_ERR      ;BRANCH IF ERROR
  011F    30         01C4   351          BSBW    LEX_SKIP_COMMA          ;SKIP THE COMMA
 32 50    E9         01C7   352          BLBC    R0,LEXICAL_ARG_ERR      ;BRANCH IF ERROR
  00B1    30         01CA   353          BSBW    GET_LEX_STRING          ;GET THE STRING TO LOCATE IN
 2C 50    E9         01CD   354          BLBC    R0,LEXICAL_ARG_ERR      ;BRANCH IF ERROR
  FE2D'   30         01D0   355          BSBW    MAC$SKIPSP              ;SKIP SPACES
29   5A    91         01D3   356          CMPB    R10,#^A/)/              ;GET TO CLOSE PAREN?
     10    13         01D6   357          BEQL    20$                     ;IF EQL YES--NO STARTPOS ARGUMENT
2C   5A    91         01D8   358          CMPB    R10,#^A/,/              ;NO--STOP ON A COMMA?
     09    12         01DB   359          BNEQ    10$                     ;IF NEQ NO--ERROR
  FE20'    30         01DD   360          BSBW    MAC$GETCHR              ;YES--GET NEXT CHARACTER
  005C     30         01E0   361          BSBW    GET_LEX_NUM             ;GET STARTPOS ARGUMENT
 0C 50     E8         01E3   362          BLBS    R0,30$                  ;IF LBS OK
     14    11         01E6   363 10$:     BRB     LEXICAL_ARG_ERR         ;GO TO ERROR ROUTINE
5A   EC 8F  90         01E8   364 20$:     MOVB    #MTX$_NOMORE,R10        ;NO STARTPOS--FLAG IT
  0130     30         01EC   365          BSBW    MAC$VSTO_CHAR           ;STORE IT AWAY
5A   29     90         01EF   366          MOVB    #^A/)/,R10              ;RESET THE PAREN
     22     11         01F2   367 30$:     BRB     LEXICAL_END_CK          ;FINISH UP
```

```
                        01F4   369          .SBTTL  PROCESS %LENGTH LEXICAL OPERATOR
                        01F4   370
                        01F4   371 ;++
                        01F4   372 ; FUNCTIONAL DESCRIPTION:
                        01F4   373 ;
                        01F4   374 ;       THIS ROUTINE PROCESSES THE %LENGTH LEXICAL OPERATOR.  THE
                        01F4   375 ;       ARGUMENT TO THE OPERATOR IS EITHER A STRING OR A DUMMY
                        01F4   376 ;       ARGUMENT NAME.
                        01F4   377 ;
                        01F4   378 ;--
                        01F4   379
                        01F4   380 LEXICAL_LENGTH:
 0087    30             01F4   381          BSBW    GET_LEX_STRING          ;READ STRING
02 50    E9             01F7   382          BLBC    R0,10$                  ;BRANCH IF THERE WAS AN ERROR
   1A    11             01FA   383          BRB     LEXICAL_END_CK          ;FINISH LEXICAL OPERATOR PROCESSING
                        01FC   384 10$:
                        01FC   385 LEXICAL_ARG_ERR:
                        01FC   386          $INTOUT_X INT$_CHKL             ;ALIGN SOURCE AND LISTING
                        0202   387          $MAC_ERR BADLEXARG              ; Get the message code
FDF6'    30             0207   388          BSBW    MAC$ERRORLN             ;ISSUE TO PASS 2
   5A    DD             020A   389          PUSHL   R10                     ;SAVE R10
   5A    D4             020C   390          CLRL    R10                     ;SEND A ZERO TO CANCEL
010E     30             020E   391          BSBW    MAC$VSTO_CHAR           ;STORE IT
   5A  8ED0             0211   392          POPL    R10                     ;RESTORE R10
   16    11             0214   393          BRB     FIND_LEX_END            ;SCAN TO END OF LEXICAL OPERATOR NOW
                        0216   394
                        0216   395 ;++
                        0216   396 ; FUNCTIONAL DESCRIPTION:
                        0216   397 ;
                        0216   398 ;       THIS ROUTINE CHECKS THAT THE LEXICAL OPERATOR WAS TERMINATED
                        0216   399 ;       PROPERLY AND ISSUES AN ERROR TO PASS 2 IF IT WAS NOT.
                        0216   400 ;
                        0216   401 ;--
                        0216   402
                        0216   403 LEXICAL_END_CK:
  FDE7'    30             0216   404          BSBW    MAC$SKIPSP              ;SKIP SPACES
29   5A    91             0219   405          CMPB    R10,#^A/)/              ;DID WE END PROPERLY?
     0E    13             021C   406          BEQL    FIND_LEX_END            ;IF EQL YES
                        021E   407          $INTOUT_X INT$_CHKL             ;NO--ALIGN SOURCE AND LISTING
                        0224   408          $MAC_ERR BADLEXFORM             ; Get message code
  FDD4'    30             0229   409          BSBW    MAC$ERRORLN             ;ISSUE ERROR TO PASS 2
                        022C   410
                        022C   411 FIND_LEX_END:
0D   5A    91             022C   412          CMPB    R10,#CR                 ;END OF LINE?
     0D    13             022F   413          BEQL    LEXICAL_EXIT            ;IF EQL YES
29   5A    91             0231   414          CMPB    R10,#^A7)/              ;NO--END OF OPERATOR?
     05    13             0234   415          BEQL    20$                     ;IF EQL YES
  FDC7'    30             0236   416          BSBW    MAC$GETCHR              ;NO--GET NEXT CHARACTER
     F1    11             0239   417          BRB     FIND_LEX_END            ;KEEP LOOKING
  FDC2'    30             023B   418 20$:     BSBW    MAC$GETCHR              ;GET CHARACTER AFTER RIGHT PAREN
                        023E   419 LEXICAL_EXIT:
           05             023E   420          RSB
```

```
                     023F    422          .SBTTL  SCAN LEXICAL OPERATOR ARGUMENT
                     023F    423
                     023F    424 ;++
                     023F    425 ; FUNCTIONAL DESCRIPTION:
                     023F    426 ;
                     023F    427 ;       THIS ROUTINE IS CALLED TO SCAN ONE LEXICAL OPERATOR ARGUMENT.
                     023F    428 ;       THE POSSIBLE ARGUMENTS ARE:
                     023F    429 ;
                     023F    430 ;       STRING ARGUMENTS:
                     023F    431 ;
                     023F    432 ;               <STRING_ENCLOSED_IN_ANGLE_BRACKETS>
                     023F    433 ;               ^@STRING_ENCLOSED_WITHIN_DELIMITERS_(@_IN_THIS_CASE)@
                     023F    434 ;               DUMMY_ARG_NAME
                     023F    435 ;
                     023F    436 ;       NUMERIC ARGUMENTS:
                     023F    437 ;
                     023F    438 ;               DECIMAL_NUMBER
                     023F    439 ;               ABSOLUTE_SYMBOL_NAME
                     023F    440 ;
                     023F    441 ; INPUTS:
                     023F    442 ;
                     023F    443 ;       12(AP)          0 OR POINTER TO MACRO ARG NAMES (MAB'S)
                     023F    444 ;
                     023F    445 ; OUTPUTS:
                     023F    446 ;
                     023F    447 ;       R0              0       ERROR ENCOUNTERED
                     023F    448 ;                       1       OK
                     023F    449 ;--
                     023F    450
                     023F    451
                     023F    452          .ENABL  LSB
                     023F    453 GET_LEX_NUM:                                ;READ A NUMBER
      FDBE'  30      023F    454          BSBW    MAC$SKIPSP                 ;SKIP ANY SPACES
        04   E0      0242    455          BBS     #CHR$V_NUM_BER,-           ;BRANCH IF IT IS A NUMBER
1D 0000'CA           0244    456                  W^MAC$AB_CHSK_TAB(R10),30$ ;...
      FDB5'  30      0248    457          BSBW    MAC$SYMSCNUP               ;NO--TRY TO SCAN ABS_SYM_NAME
     03 50   E8      024B    458          BLBS    R0,10$                     ;IF LBS GOT SYMBOL NAME
                     024E    459 ;
                     024E    460 ; NUMBER EXPECTED, DID NOT FIND EITHER A DECIMAL_NUMBER OR AN
                     024E    461 ; ABSOLUTE_SYMBOL_NAME.
                     024E    462 ;
        50   D4      024E    463          CLRL    R0                         ;RETURN FAILURE
             05      0250    464          RSB
      FDAC'  30      0251    465 10$:     BSBW    MAC$INSUSRSYMTB            ;FOUND SYMBOL NAME--LOOK UP OR ENTER
                     0254    466                                             ;IN USER SYMBOL TABLE
        51   DD      0254    467          PUSHL   R1                         ;SAVE SYMBOL BLOCK ADDRESS
   50   05   D0      0256    468          MOVL    #5,R0                      ;SET TO STORE 5 BYTES
     00E9   30      0259    469          BSBW    MAC$VSTO_SET               ;PREPARE TO STORE THEM
88   ED 8F   90      025C    470          MOVB    #MTX$_SYMADR,(R8)+         ;FIRST BYTE IS MACRO OPERATOR
                     0260    471                                             ;FLAGGING SYMBOL BLOCK ADDRESS
        88 8ED0      0260    472          POPL    (R8)+                      ;STORE THE SYMBOL BLOCK ADDRESS
        7A   11      0263    473          BRB     300$                       ;RETURN WITH SUCCESS
                     0265    474 ;
                     0265    475 ; EXPECTING DECIMAL_NUMBER, AND FOUND A DIGIT--READ THE NUMBER IN
                     0265    476 ;
        58   DD      0265    477 30$:     PUSHL   R8                         ;SAVE R8, WHICH DNUMBER WIPES
      FD96'  30      0267    478          BSBW    MAC$DNUMBER                ;READ THE NUMBER
```

```
                58 8ED0  026A   479          POPL    R8                      ;RESTORE R8
             50   05    D0  026D   480          MOVL    #5,R0                   ;SET TO STORE 5 BYTES
                00D2    30  0270   481          BSBW    MAC$VSTO_SET            ;PREPARE TO STORE THEM
          88    EE 8F    90  0273   482          MOVB    #MTX$_LITVAL,(R8)+      ;FLAG LITERAL VALUE FOLLOWS
       88   0000'CF    D0  0277   483          MOVL    W^MAC$GL_VALUE,(R8)+    ;STORE THE VALUE
                61    11  027C   484          BRB     300$                    ;RETURN WITH SUCCESS
                          027E   485 ;
                          027E   486 ; STRING IS EXPECTED
                          027E   487 ;
                          027E   488 GET_LEX_STRING:
                FD7F'     30  027E   489          BSBW    MAC$SKIPSP              ;SKIP SPACES
                   02     E0  0281   490          BBS     #CHR$V_SYM_CHR,-        ;BRANCH IF FORMAL ARG NAME
          36 0000'CA          0283   491                  W^MAC$AB_CHSK_TAB(R10),200$ ;
                          0287   492 ;
                          0287   493 ; MUST BE EITHER A STRING IN ANGLE BRACKETS OR AN UPARROW-DELIMITER
                          0287   494 ; TYPE STRING.
                          0287   495 ;
             5E 8F    5A    91  0287   496          CMPB    R10,#^A/^/              ;IS IT THE UPARROW?
                08    13  028B   497          BEQL    120$                    ;IF EQL YES
                3C    5A    91  028D   498          CMPB    R10,#^A/</              ;NO--ANGLE BRACKET?
                03    13  0290   499          BEQL    120$                    ;IF EQL YES
                50    D4  0292   500          CLRL    R0                      ;RETURN ERROR
                      05  0294   501          RSB
             00 6B    21    E3  0295   502 120$:   BBCS    #FLG$V_LEXOP,(R11),.+1  ;FLAG LEXICAL OPERATOR CALL
                FD64'     30  0299   503          BSBW    MAC$MAC_ARG_SCN         ;SCAN THE MACRO ARGUMENT
             00 6B    21    E5  029C   504          BBCC    #FLG$V_LEXOP,(R11),.+1  ;CLEAR LEXICAL OPERATOR FLAG
                50    DD  02A0   505          PUSHL   R0                      ;SAVE LENGTH OF ARGUMENT SCANNED
             50   03    C0  02A2   506          ADDL2   #3,R0                   ;FIGURE LENGTH NEEDED
                          02A5   507                                          ;(MTX$_LITSTR AND LENGTH WORD)
                009D    30  02A5   508          BSBW    MAC$VSTO_SET            ;SET TO STORE THE STRING
          88    EF 8F    90  02A8   509          MOVB    #MTX$_LITSTR,(R8)+      ;SET INDICATOR BYTE
                50 8ED0  02AC   510          POPL    R0                      ;GET THE STRING COUNT BACK
             88   50    B0  02AF   511          MOVW    R0,(R8)+                ;STORE THE COUNT WORD
       68   0000'CF    50  28  02B2   512          MOVC3   R0,W^MAC$AB_TMPBUF,(R8) ;COPY THE ARGUMENT OUT
                58    53  D0  02B8   513          MOVL    R3,R8                   ;UPDATE THE POINTER
                22    11  02BB   514          BRB     300$                    ;RETURN WITH SUCCESS
                          02BD   515 ;
                          02BD   516 ; IT APPEARS WE WILL BE SCANNING A DUMMY ARGUMENT NAME
                          02BD   517 ;
                FD40'     30  02BD   518 200$:   BSBW    MAC$SYMSCNUP            ;SCAN THE DUMMY ARG NAME
             55   0C AC    D0  02C0   519          MOVL    12(AP),R5               ;POINT TO MACRO ARG BLOCK LIST
                1D    13  02C4   520          BEQL    310$                    ;IF EQL NO ARGS--OOPS
                FD37'     30  02C6   521          BSBW    MAC$SRC_LIST            ;LOOK UP THE DUMMY ARG
                17 50     E9  02C9   522          BLBC    R0,310$                 ;BRANCH IF NOT FOUND
                51    DD  02CC   523          PUSHL   R1                      ;SAVE MAB ADDRESS
             50   02    90  02CE   524          MOVB    #2,R0                   ;SET TO STORE 2 BYTES
                0071    30  02D1   525          BSBW    MAC$VSTO_SET            ;SET UP FOR THEM
          88    FF 8F    90  02D4   526          MOVB    #MTX$_ARGMRK,(R8)+      ;STORE ARG MARKER BYTE
                51 8ED0  02D8   527          POPL    R1                      ;RESTORE MAB ADDRESS
          88    05 A1    90  02DB   528          MOVB    MAB$B_ARGNO(R1),(R8)+   ;STORE ARG NUMBER
                          02DF   529 ;
                          02DF   530 ; COME HERE TO SET SUCCESS AND RETURN
                          02DF   531 ;
             50   01    D0  02DF   532 300$:   MOVL    #1,R0                   ;FLAG GOOD ARGUMENT
                      05  02E2   533          RSB                             ;ALL DONE
                50    D4  02E3   534 310$:   CLRL    R0                      ;FLAG AN ERROR
                      05  02E5   535          RSB
```

```
                02E6    536             .DSABL  LSB
```

```
                     02E6    538 ;++
                     02E6    539 ; FUNCTIONAL DESCRIPTION:
                     02E6    540 ;
                     02E6    541 ;       THIS ROUTINE SKIPS SPACES, EXPECTS A COMMA THEN, AND RETURNS
                     02E6    542 ;       WITH THE CHARACTER FOLLOWING THE COMMA.
                     02E6    543 ;
                     02E6    544 ;--
                     02E6    545
                     02E6    546 LEX_SKIP_COMMA:
   FD17'  30         02E6    547         BSBW    MAC$SKIPSP              ;SKIP SPACES
2C   5A  91         02E9    548         CMPB    R10,#^A/,/              ;DID WE GET TO A COMMA?
     03  13         02EC    549         BEQL    10$                     ;IF EQL YES
     50  D4         02EE    550         CLRL    R0                      ;NO--FLAG AN ERROR
         05         02F0    551         RSB
   FD0C'  30         02F1    552 10$:    BSBW    MAC$GETCHR              ;READ CHARACTER AFTER COMMA
50   01  D0         02F4    553         MOVL    #1,R0                   ;FLAG SUCCESS
         05         02F7    554         RSB
```

```
                     02F8    556          .SBTTL  SPECIAL DIRECTIVES SCANNED DURING BODY_SCAN
                     02F8    557
                     02F8    558 ;++
                     02F8    559 ; FUNCTIONAL DESCRIPTION:
                     02F8    560 ;
                     02F8    561 ;       THIS ROUTINE IS CALLED WHEN A .REPT, .IRP, .IRPC, OR
                     02F8    562 ;       .MACRO IS SCANNED WHILE READING THE MACRO BODY INTO
                     02F8    563 ;       CORE.  THE MACRO DEFINITION LEVEL IS INCREMENTED SO
                     02F8    564 ;       THAT BODY_SCAN KNOWS WHEN THE PROPER .ENDM HAS BEEN
                     02F8    565 ;       FOUND.
                     02F8    566 ;
                     02F8    567 ;--
                     02F8    568
                     02F8    569 MAC$MAC_IN_MAC::
 0000'CF    D6       02F8    570          INCL    W^MAC$GL_MCLVL          ;BUMP THE MACRO DEFINITION LEVEL
            05       02FC    571          RSB
                     02FD    572
                     02FD    573 ;++
                     02FD    574 ; FUNCTIONAL DESCRIPTION:
                     02FD    575 ;
                     02FD    576 ;       THIS ROUTINE IS CALLED WHEN A .END, .ENDM, OR .ENDR IS
                     02FD    577 ;       SCANNED IN BODY_SCAN.  THE MACRO DEFINITION LEVEL IS
                     02FD    578 ;       DECREMENTED, AND IF MACRO DEFINITIONS ARE NOT BEING
                     02FD    579 ;       LISTED, THE LIST_IT FLAG IS CLEARED.
                     02FD    580 ;
                     02FD    581 ;--
                     02FD    582
                     02FD    583 MAC$MAC_DEF_END::
 0000'CF    D7       02FD    584          DECL    W^MAC$GL_MCLVL          ;DECREMENT MACRO DEF. LEVEL
      16    14       0301    585          BGTR    10$                     ;IF GTR STILL IN A MACRO
    FCFA'   30       0303    586          BSBW    MAC$CREF_DIR            ;CREF DIRECTIVE IF CREFFING
                     0306    587                                          ;(R1 STILL POINTS TO SYMBOL BLOCK)
0E 0005'CF   E8      0306    588          BLBS    W^LST$G_MACRODEF+SYM$L_VAL,10$ ;BRANCH IF LISTING MACRO
                     030B    589                                                 ;DEFS
                     030B    590          $INTOUT_LW INT$_SETLONG,<#0,#MAC$GL_LIST_IT> ;NO--OUTPUT FLAG
            05       0319    591 10$:     RSB
                     031A    592
                     031A    593 ;++
                     031A    594 ; FUNCTIONAL DESCRIPTION:
                     031A    595 ;
                     031A    596 ;       THIS ROUTINE IS CALLED WHEN CERTAIN DIRECTIVES (.ERROR, .WARN,
                     031A    597 ;       .PRINT, .ASCIX) ARE SCANNED.  THESE DIRECTIVES ARE SPECIAL IN
                     031A    598 ;       THAT THEY MAY HAVE SEMI-COLONS AS PART OF THEIR SYNTAX.  THEREFORE
                     031A    599 ;       THE SPECIAL ALL CHARACTERS FLAG (SPLALL) IS SET SO THAT THESE
                     031A    600 ;       SEMI-COLONS ARE NOT TAKEN AS COMMENTS.
                     031A    601 ;
                     031A    602 ;--
                     031A    603
                     031A    604 MAC$SET_ALL_CHR::
00 6B    1A          031A    605          BBCS    #FLG$V_SPLALL,(R11),.+1 ;SET SPECIAL ALL CHARACTERS FLAG
            05       031E    606          RSB                             ;THAT'S ALL
```

```
                     031F   608          .SBTTL  STORE CHARACTER INTO VIRTUAL MEMORY
                     031F   609
                     031F   610 ;++
                     031F   611 ; FUNCTIONAL DESCRIPTION:
                     031F   612 ;
                     031F   613 ;       THIS ROUTINE STORES THE CHARACTER IN R10 INTO VIRTUAL MEMORY.
                     031F   614 ;
                     031F   615 ; INPUTS:
                     031F   616 ;
                     031F   617 ;       R8      PTR INTO VIRTUAL MEMORY
                     031F   618 ;       R7      COUNT OF BYTES LEFT IN THIS BLOCK OF VIRT. MEMORY
                     031F   619 ;       R6      COUNT OF CHARACTERS STORED
                     031F   620 ;
                     031F   621 ; OUTPUTS:
                     031F   622 ;
                     031F   623 ;       R8      UPDATED
                     031F   624 ;       R7      UPDATED
                     031F   625 ;       R6      UPDATED
                     031F   626 ;
                     031F   627 ;       CHARACTER IN R10 STORED INTO STRING
                     031F   628 ;
                     031F   629 ;--
                     031F   630
                     031F   631 MAC$VSTO_CHAR::
50   57   07   C3    031F   632          SUBL3   #7,R7,R0                ;SEE IF ROOM FOR 2 CHARACTERS AND LINK
          02   14    0323   633          BGTR    10$                     ;IF GT YES
          2F   10    0325   634          BSBB    MAC$VSTO_GET            ;NO--GET A NEW PAGE
     88   5A   90    0327   635 10$:     MOVB    R10,(R8)+               ;STORE CHAR IN VM
          56   D6    032A   636          INCL    R6                      ;COUNT CHARACTERS STORED
          57   D7    032C   637          DECL    R7                      ;DEC. COUNT OF REMAINING SPACE
               05    032E   638          RSB                             ;DONE
```

```
                     032F   640          .SBTTL  STORE STRING INTO VIRTUAL MEMORY
                     032F   641
                     032F   642 ;+
                     032F   643 ; FUNCTIONAL DESCRIPTION:
                     032F   644 ;
                     032F   645 ;         THIS ROUTINE STORES A STRING INTO VIRTUAL MEMORY.
                     032F   646 ;
                     032F   647 ; INPUTS:
                     032F   648 ;
                     032F   649 ;         R2      ADDRESS OF THE STRING
                     032F   650 ;         R3      LENGTH OF THE STRING
                     032F   651 ;         R6      CHARS STORED IN VM
                     032F   652 ;         R7      CHARS LEFT IN VM
                     032F   653 ;         R8      POINTER INTO VM
                     032F   654 ;
                     032F   655 ; OUTPUTS:
                     032F   656 ;
                     032F   657 ;         R6,R7,R8 ARE UPDATED
                     032F   658 ;         STRING IS STORED
                     032F   659 ;
                     032F   660 ;--
                     032F   661
                     032F   662 MAC$VSTO_STR::
50   57   53   C3    032F   663          SUBL3   R3,R7,R0                ;SEE IF ROOM FOR STRING
               02    14  0333   664          BGTR    10$                     ;IF GTR YES
               1F    10  0335   665          BSBB    MAC$VSTO_GET            ;NO--GET A PAGE
     57   53   C2    0337   666 10$:       SUBL2   R3,R7                   ;DECREASE COUNT LEFT
     56   53   C0    033A   667          ADDL2   R3,R6                   ;INCREASE COUNT STORED
68   62   53   28    033D   668          MOVC3   R3,(R2),(R8)            ;COPY INTO BUFFER
     58   53   D0    0341   669          MOVL    R3,R8                   ;UPDATE THE VM POINTER
               05    0344   670          RSB                             ;DONE
```

```
                     0345   672          .SBTTL  SETUP TO STORE INTO VIRTUAL MEMORY
                     0345   673
                     0345   674 ;++
                     0345   675 ; FUNCTIONAL DESCRIPTION:
                     0345   676 ;
                     0345   677 ;       THIS ROUTINE PREPARES TO STORE 'N' BYTES IN VM.
                     0345   678 ;
                     0345   679 ; INPUTS:
                     0345   680 ;
                     0345   681 ;       R0      NUMBER OF BYTES TO STORE
                     0345   682 ;       R6      COUNT OF BYTES STORED
                     0345   683 ;       R7      COUNT OF BYTES LEFT
                     0345   684 ;       R8      POINTER INTO VM
                     0345   685 ;
                     0345   686 ; OUTPUTS:
                     0345   687 ;
                     0345   688 ;       R6,R7   UPDATED SO ALL CALLER NEEDS TO DO IS STORE BYTES (R8)
                     0345   689 ;
                     0345   690 ;--
                     0345   691
                     0345   692 MAC$VSTO_SET::
          50   DD    0345   693          PUSHL   R0                      ;SAVE THE COUNT
50   57   50   C3    0347   694          SUBL3   R0,R7,R0                ;SEE IF ROOM
          02   14    034B   695          BGTR    10$                     ;IF GTR YES
          07   10    034D   696          BSBB    MAC$VSTO_GET            ;NO--GET A PAGE
     57   6E   C2    034F   697 10$:     SUBL2   (SP),R7                 ;ADJUST THE COUNT LEFT
     56   8E   C0    0352   698          ADDL2   (SP)+,R6                ;ADJUST COUNT STORED
               05    0355   699          RSB                             ;BACK TO CALLER
```

```
                     0356   701          .SBTTL  ALLOCATE VIRTUAL BUFFER PAGE
                     0356   702
                     0356   703 ;++
                     0356   704 ; FUNCTIONAL DESCRIPTION:
                     0356   705 ;
                     0356   706 ;       THIS ROUTINE ALLOCATES A PAGE OF VIRTUAL MEMORY AND LINKS THE
                     0356   707 ;       TEXT STREAM ON THE CURRENT PAGE TO THE NEW PAGE.  THE POINTER
                     0356   708 ;       (R8) AND COUNTER (R7) ARE UPDATED.
                     0356   709 ;
                     0356   710 ; INPUTS:
                     0356   711 ;
                     0356   712 ;       R8      POINTER TO CURRENT PAGE (WHERE WE ARE STORING)
                     0356   713 ;       R7      COUNT  OF BYTES LEFT ON PAGE
                     0356   714 ;       MAC$GL_MACPTR   POINTER TO MNB FOR MACRO BEING DEFINED
                     0356   715 ;
                     0356   716 ; OUTPUTS:
                     0356   717 ;
                     0356   718 ;       R8      POINTS TO FIRST BYTE ON NEW PAGE
                     0356   719 ;       R7      Number of usable bytes in page (512 bytes/page -
                     0356   720 ;               MXB$K_BLKSIZ for header and 6 bytes for link byte flag)
                     0356   721 ;
                     0356   722 ;--
                     0356   723
                     0356   724 MAC$VSTO_GET::
        FCA7'  30    0356   725          BSBW    MAC$ALL_1_PAGE          ;ALLOCATE A PAGE (ADDR IN R0 ON RETURN)
  88    FE 8F  90    0359   726          MOVB    #MTX$_TXTLNK,(R8)+      ;STORE TEXT LINK FLAG BYTE
  00 6B    22  E2    035D   727          BBSS    #FLG$V_SPECOP,(R11),.+1 ;FLAG SPECIAL OPERATOR ON LINE
                     0361   728 MAC$VSTO_GET_0:                          ;ENTRY TO SET LINKS ONLY
      88    50  D0    0361   729          MOVL    R0,(R8)+                ;STORE TEXT LINK LINK POINTER
      58    50  D0    0364   730          MOVL    R0,R8                   ;POINT R8 TO THE NEW PAGE
57    01F2  8F  3C    0367   731          MOVZWL  #<512-MXB$K_BLKSIZ-6>,R7 ; Set byte count
51    0000'CF   D0    036C   732          MOVL    W^MAC$GL_MACPTR,R1      ;GET POINTER TO MNB FOR MACRO
                     0371   733          ASSUME MXB$L_LINK  EQ 0
                     0371   734          ASSUME MXB$L_PAGES EQ MXB$L_LINK+4
  88    0B A1  D0    0371   735          MOVL    MNB$L_PAGP(R1),(R8)+    ;LINK NEW PAGE INTO PAGE LIST
  0B A1    50  D0    0375   736          MOVL    R0,MNB$L_PAGP(R1)       ;...
      88    01  D0    0379   737          MOVL    #1,(R8)+                ; Block size is 1 page
        0F A1  D6    037C   738          INCL    MNB$L_PAGC(R1)          ;AND COUNT IT
  04 6B    1C  E0    037F   739          BBS     #FLG$V_RPTIRP,(R11),10$ ;BRANCH IF RPTIRP (NOT MACRO)
      0000'CF   D6    0383   740          INCL    W^MAC$GL_MCPGRQ         ;COUNT ANOTHER PAGE FOR MACRO DEFINITION
                05    0387   741 10$:     RSB                             ;RETURN TO CALLER
                     0388   742
                     0388   743          .END
```

```
$COUNT            = 0000003B
ARG$K_SIZE        = 000003E8
AUD$K_SIZE        = 00000010
BLNK              = 00000020
BODY_SCAN_END       0000011C R      04
BODY_SCAN_LOOP      0000001E R      04
CHR$M_COMMA_CR    = 00000020
CHR$M_ILL_CHR     = 00000040
CHR$M_NUM_BER     = 00000010
CHR$M_SPA_MSK     = 00000001
CHR$M_SYM_CH1     = 00000008
CHR$M_SYM_CHR     = 00000004
CHR$M_SYM_DLM     = 00000002
CHR$V_COMMA_CR    = 00000005
CHR$V_CVTLWC      = 00000061
CHR$V_ILL_CHR     = 00000006
CHR$V_NOCVT       = 0000007F
CHR$V_NUM_BER     = 00000004
CHR$V_SPA_MSK     = 00000000
CHR$V_SYM_CH1     = 00000003
CHR$V_SYM_CHR     = 00000002
CHR$V_SYM_DLM     = 00000001
CNT               = 00000002
CR                = 0000000D
ERR               = 00000000
FF                = 0000000C
FIND_LEX_END        0000022C R      04
FLG$M_ALLCHR      = 00000001
FLG$M_BOL         = 00000002
FLG$M_CHKLPND     = 00100000
FLG$M_COMPEXPR    = 00000004
FLG$M_CONT        = 00000008
FLG$M_CRF         = 40000000
FLG$M_CRSEEN      = 00000001
FLG$M_DATRPT      = 00000010
FLG$M_DBGOUT      = 00004000
FLG$M_DLIMSTR     = 00008000
FLG$M_ENDMCH      = 00000020
FLG$M_EVALEXPR    = 00000040
FLG$M_EXPOPT      = 00000080
FLG$M_EXTERR      = 00010000
FLG$M_EXTWRN      = 00020000
FLG$M_FIRSTLN     = 00000200
FLG$M_IFSTAT      = 00800000
FLG$M_IIF         = 00400000
FLG$M_INSERT      = 00000100
FLG$M_IRPC        = 20000000
FLG$M_LEXOP       = 00000002
FLG$M_LSTXST      = 00000200
FLG$M_MAC2COL     = 00000800
FLG$M_MACL        = 00000800
FLG$M_MACLTB      = 08000000
FLG$M_MACTXT      = 00010000
FLG$M_MEBLST      = 00001000
FLG$M_MOREARG     = 00002000
FLG$M_MOREINP     = 00000008
FLG$M_NEWPND      = 00000400
FLG$M_NOREF       = 01000000
FLG$M_NTYPEPC     = 00000020
FLG$M_NULCHR      = 00040000
FLG$M_OBJXST      = 00200000
FLG$M_OPNDCHK     = 00000100
FLG$M_OPRND       = 00002000
FLG$M_OPTVFLIDX   = 00001000
FLG$M_ORDLST      = 00020000
FLG$M_P2          = 00004000
FLG$M_RPTIRP      = 10000000
FLG$M_SEQFIL      = 02000000
FLG$M_SKAN        = 00008000
FLG$M_SPECOP      = 00000004
FLG$M_SPLALL      = 04000000
FLG$M_STOIMF      = 00040000
FLG$M_SYM2COL     = 00000400
FLG$M_TOCFLG      = 00080000
FLG$M_UPAFLG      = 00000010
FLG$M_UPDFIL      = 00000080
FLG$M_UPMARG      = 00000040
FLG$M_XCRF        = 80000000
FLG$V_ALLCHR      = 00000000
FLG$V_BOL         = 00000001
FLG$V_CHKLPND     = 00000014
FLG$V_COMPEXPR    = 00000002
FLG$V_CONT        = 00000003
FLG$V_CRF         = 0000001E
FLG$V_CRSEEN      = 00000020
FLG$V_DATRPT      = 00000004
FLG$V_DBGOUT      = 0000002E
FLG$V_DLIMSTR     = 0000002F
FLG$V_ENDMCH      = 00000005
FLG$V_EVALEXPR    = 00000006
FLG$V_EXPOPT      = 00000007
FLG$V_EXTERR      = 00000030
FLG$V_EXTWRN      = 00000031
FLG$V_FIRSTLN     = 00000029
FLG$V_IFSTAT      = 00000017
FLG$V_IIF         = 00000016
FLG$V_INSERT      = 00000008
FLG$V_IRPC        = 0000001D
FLG$V_LEXOP       = 00000021
FLG$V_LSTXST      = 00000009
FLG$V_MAC2COL     = 0000002B
FLG$V_MACL        = 0000000B
FLG$V_MACLTB      = 0000001B
FLG$V_MACTXT      = 00000010
FLG$V_MEBLST      = 0000000C
FLG$V_MOREARG     = 0000002D
FLG$V_MOREINP     = 00000023
FLG$V_NEWPND      = 0000000A
FLG$V_NOREF       = 00000018
FLG$V_NTYPEPC     = 00000025
FLG$V_NULCHR      = 00000032
FLG$V_OBJXST      = 00000015
FLG$V_OPNDCHK     = 00000028
FLG$V_OPRND       = 0000000D
FLG$V_OPTVFLIDX   = 0000002C
FLG$V_ORDLST      = 00000011
FLG$V_P2          = 0000000E
FLG$V_RPTIRP      = 0000001C
FLG$V_SEQFIL      = 00000019
FLG$V_SKAN        = 0000000F
FLG$V_SPECOP      = 00000022
FLG$V_SPLALL      = 0000001A
FLG$V_STOIMF      = 00000012
FLG$V_SYM2COL     = 0000002A
FLG$V_TOCFLG      = 00000013
FLG$V_UPAFLG      = 00000024
FLG$V_UPDFIL      = 00000027
FLG$V_UPMARG      = 00000026
FLG$V_XCRF        = 0000001F
GET_LEX_NUM         0000025F R      04
GET_LEX_STRING      0000027E R      04
HASASZ            = 0000007F
HYPHEN            = 0000002D
INP$K_BUFSIZ      = 000003E8
INSYMC            = 00000006
INSYMP            = 00000030 R    03
INSYTM            = 00000030 R    03
INT$K_BUFSIZ      = 000013F4
INT$K_BUFWRN      = 00001390
INT$_ADD          = 00000001
INT$_AND          = 00000002
INT$_ASH          = 00000003
INT$_ASN          = 0000000C
INT$_AUGPC        = 0000000D
INT$_BDST         = 0000000E
INT$_CHKL         = 0000000F
INT$_DIV          = 00000004
INT$_END          = 00000010
INT$_EPT          = 00000011
INT$_ERR          = 00000012
INT$_ETX          = 00000013
INT$_FNEWL        = 00000014
INT$_ILG          = 00000000
INT$_INFO         = 0000003A
INT$_LGLAB        = 00000015
INT$_MACL         = 00000016
INT$_MUL          = 00000005
INT$_NEG          = 00000006
INT$_NEWL         = 00000017
INT$_NEWP         = 00000018
INT$_NOT          = 00000007
INT$_OP           = 00000019
INT$_OR           = 00000008
INT$_PRIL         = 0000001A
INT$_PRT          = 0000001B
INT$_PSECT        = 0000001C
INT$_REDEF        = 0000001D
INT$_REF          = 0000001E
INT$_REST         = 0000001F
INT$_SAME         = 00000009
INT$_SAVE         = 00000020
```

```
INT$_SBTTL      = 00000021
INT$_SETFLAG    = 00000022
INT$_SETLONG    = 00000023
INT$_SPIC       = 00000024
INT$_SPID       = 00000025
INT$_STIB       = 00000026
INT$_STIL       = 00000028
INT$_STIW       = 00000027
INT$_STKEPT     = 00000029
INT$_STKG       = 0000002A
INT$_STKL       = 0000002B
INT$_STKPC      = 0000002C
INT$_STKS       = 0000002D
INT$_STOB       = 00000034
INT$_STOL       = 0000002E
INT$_STOW       = 00000035
INT$_STRB       = 0000002F
INT$_STRL       = 00000031
INT$_STRSB      = 00000032
INT$_STRSW      = 00000033
INT$_STRW       = 00000030
INT$_STSB       = 00000036
INT$_STSW       = 00000037
INT$_SUB        = 0000000A
INT$_SUME       = 00000039
INT$_WRN        = 00000038
INT$_XOR        = 0000000B
LEXICAL_ARG_ERR   000001FC R      04
LEXICAL_END_CK    00000216 R      04
LEXICAL_EXIT      0000023E R      04
LEXICAL_EXTRACT   000001A1 R      04
LEXICAL_LENGTH    000001F4 R      04
LEXICAL_LOCATE    000001BE R      04
LEX_SKIP_COMMA    000002E6 R      04
LST$G_MACRODEF    ********   X    04
LST$K_BUFSIZ    = 00000086
LST$K_L_P_PAGE  = 0000003C
LST$K_TITLE_SIZ = 00000028
MAB$B_ARGNO       00000005
MAB$B_NAME        00000004
MAB$K_BLKSIZ      0000000C
MAB$L_DVPTR       00000008
MAB$L_LINK        00000000
MAB$W_DVLEN       00000006
MAC$AB_CMSK_TAB   ********   X    04
MAC$AB_LINEBF     ********   X    04
MAC$AB_TMPBUF     ********   X    04
MAC$AB_TMPSY1     ********   X    04
MAC$AB_TMPSYM     ********   X    04
MAC$ALL_1_PAGE    ********   X    04
MAC$BODY_SCAN     00000000 RG      04
MAC$CREF_DIR      ********   X    04
MAC$DNUMBER       ********   X    04
MAC$ERRORLN       ********   X    04
MAC$GETCHR        ********   X    04
MAC$GL_ARGPTR     ********   X    04
MAC$GL_LINEPT     ********   X    04
MAC$GL_LIST_IT    ********   X    04
MAC$GL_MACPTR     ********   X    04
MAC$GL_MCLVL      ********   X    04
MAC$GL_MCPGRQ     ********   X    04
MAC$GL_VALUE      ********   X    04
MAC$G_SPLMC_DIR   ********   X    04
MAC$INSUSRSYMTB   ********   X    04
MAC$INTOUT_2_LW   ********   X    04
MAC$INTOUT_X      ********   X    04
MAC$LEX_OP_LIST   00000030 RG      03
MAC$MAC_ARG_SCN   ********   X    04
MAC$MAC_DEF_END   000002FD RG      04
MAC$MAC_IN_MAC    000002F8 RG      04
MAC$SET_ALL_CHR   0000031A RG      04
MAC$SKIPSP        ********   X    04
MAC$SRC_LIST      ********   X    04
MAC$SYMSCNUP      ********   X    04
MAC$SYMSCNUP1     ********   X    04
MAC$VSTO_CHAR     0000031F RG      04
MAC$VSTO_GET      00000356 RG      04
MAC$VSTO_GET_0    00000361 R       04
MAC$VSTO_SET      00000345 RG      04
MAC$VSTO_STR      0000032F RG      04
MAC$_BADLEXARG  = 007D9032
MAC$_BADLEXFORM = 007D903A
MAC_SUBSYS      = 0000007D
MNB$B_ARGCT       00000017
MNB$B_NAME        00000004
MNB$K_BLKSIZ      0000001C
MNB$L_ARGP        00000018
MNB$L_CRSYMF      00000013
MNB$L_LINK        00000000
MNB$L_PAGC        0000000F
MNB$L_PAGP        0000000B
MNB$L_TXTP        00000005
MNB$W_FLAG        00000009
MTX$_ARGMRK     = 000000FF
MTX$_LITSTR     = 000000EF
MTX$_LITVAL     = 000000EE
MTX$_LXEXT      = 000000FC
MTX$_LXLEN      = 000000FD
MTX$_LXLOC      = 000000FB
MTX$_NOMORE     = 000000EC
MTX$_SYMADR     = 000000ED
MTX$_TXTLNK     = 000000FE
MXB$K_BLKSIZ      00000008
MXB$L_LINK        00000000
MXB$L_PAGES       00000004
OBJ$K_BUFSIZ    = 00000200
OPF$M_LASTOPR   = 00002000
OPF$M_OPTEXP    = 00001000
OPF$V_LASTOPR   = 0000000D
OPF$V_OPTEXP    = 0000000C
PSC$B_NAME        00000004
PSC$B_SEG         0000000C
PSC$B_UNUSED      0000000B
PSC$K_BLKSIZ      00000013
PSC$K_NO_OPTNS  = 0000000A
PSC$L_CURLOC      0000000F
PSC$L_LINK        00000000
PSC$L_MAXLGTH     00000005
PSC$M_ABS       = FFFFFFF7
PSC$M_ALIGNFLG  = 00004000
PSC$M_ALLOPTNS  = 000003FF
PSC$M_BYTE      = 00004000
PSC$M_CON       = FFFFFFFB
PSC$M_DEFAULT   = 000001C8
PSC$M_EXE       = 000000C0
PSC$M_GBL       = 00000010
PSC$M_LCL       = FFFFFFEF
PSC$M_LIB       = 00000002
PSC$M_LONG      = 00004800
PSC$M_NOEXE     = FFFFFFBF
PSC$M_NOPIC     = FFFFFFFE
PSC$M_NORD      = FFFFFF7F
PSC$M_NOSHR     = FFFFFFDF
PSC$M_NOVEC     = FFFFFDFF
PSC$M_NOWRT     = FFFFFEFF
PSC$M_OVR       = 00000004
PSC$M_PAGE      = 00006400
PSC$M_PIC       = 00000001
PSC$M_QUAD      = 00004C00
PSC$M_RD        = 00000080
PSC$M_REL       = 00000008
PSC$M_SHR       = 00000020
PSC$M_USR       = FFFFFFFD
PSC$M_VEC       = 00000200
PSC$M_WORD      = 00004400
PSC$M_WRT       = 00000180
PSC$S_ALIGNMENT = 00000004
PSC$V_ALIGNFLG  = 0000000E
PSC$V_ALIGNMENT = 0000000A
PSC$V_EXE       = 00000006
PSC$V_GBL       = 00000004
PSC$V_LIB       = 00000001
PSC$V_OVR       = 00000002
PSC$V_PIC       = 00000000
PSC$V_RD        = 00000007
PSC$V_REL       = 00000003
PSC$V_SHR       = 00000005
PSC$V_VEC       = 00000009
PSC$V_WRT       = 00000008
PSC$W_FLAG        00000009
PSC$W_OPTIONS     0000000D
RDX$V_BINARY    = 00000000
RDX$V_DECIMAL   = 00000002
RDX$V_DOUBLE    = 00000005
RDX$V_FLOAT     = 00000004
RDX$V_GFLOAT    = 00000006
RDX$V_HEX       = 00000003
RDX$V_HFLOAT    = 00000007
RDX$V_OCTAL     = 00000001
REG$_PC         = 0000000F
SCAN_PERCENT      00000156 R      04
```

```
SEMI              =  0000003B
STB$K_PG_MISS     =  0000000A
SYM$B_NAME           00000004
SYM$B_SEG            0000000C
SYM$B_TOKEN          0000000B
SYM$K_BLKSIZ         0000000D
SYM$K_MAXLEN      =  0000001F
SYM$K_TWOCOL      =  00000010
SYM$L_LINK           00000000
SYM$L_VAL            00000005
SYM$M_ABS         =  00000010
SYM$M_ASN         =  00000100
SYM$M_CRFO        =  00002000
SYM$M_DEBUG       =  00000020
SYM$M_DEF         =  00000001
SYM$M_DELMAC      =  00000200
SYM$M_EPT         =  00000200
SYM$M_EXTRN       =  00000008
SYM$M_GLOBL       =  00000004
SYM$M_LOCAL       =  00000040
SYM$M_ODBG        =  00000400
SYM$M_REF         =  00000080
SYM$M_RELPSECT    =  00000800
SYM$M_SUPR        =  00004000
SYM$M_WEAK        =  00000002
SYM$M_XCRF        =  00001000
SYM$V_ABS         =  00000004
SYM$V_ASN         =  00000008
SYM$V_CRFO        =  0000000D
SYM$V_DEBUG       =  00000005
SYM$V_DEF         =  00000000
SYM$V_DELMAC      =  00000009
SYM$V_EPT         =  00000009
SYM$V_EXTRN       =  00000003
SYM$V_GLOBL       =  00000002
SYM$V_LOCAL       =  00000006
SYM$V_ODBG        =  0000000A
SYM$V_REF         =  00000007
SYM$V_RELPSECT    =  0000000B
SYM$V_SUPR        =  0000000E
SYM$V_WEAK        =  00000001
SYM$V_XCRF        =  0000000C
SYM$W_FLAG           00000009
TAB               =  00000009
X1                =  00000400
X2                =  0000000F
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
. BLANK .	00000000 (0.)	01 (1.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE
ABS	0000001C (28.)	02 (2.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
MAC$RO_DATA	0000003D (61.)	03 (3.)	NOPIC USR CON REL GBL NOSHR NOEXE RD NOWRT NOVEC LONG

MAC$RO_CODE_MAC 00000388 (904.) 04 (4.) NOPIC USR CON REL GBL NOSHR EXE RD NOWRT NOVEC LONG

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	29	00:00:00.07	00:00:00.85
Command processing	104	00:00:00.34	00:00:02.62
Pass 1	224	00:00:03.80	00:00:18.03
Symbol table sort	0	00:00:00.47	00:00:01.66
Pass 2	140	00:00:01.21	00:00:05.52
Symbol table output	32	00:00:00.17	00:00:00.63
Psect synopsis output	3	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	534	00:00:06.08	00:00:29.33

The working set limit was 1500 pages.
35702 bytes (70 pages) of virtual memory were used to buffer the intermediate code.
There were 30 pages of symbol table space allocated to hold 466 non-local and 40 local symbols.
743 source lines were read in Pass 1, producing 20 object records in Pass 2.
17 pages of virtual memory were used to define 16 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[MACRO.OBJ]MACRO.MLB;1	13
_255DUA28:[SYSLIB]STARLET.MLB;2	4
TOTALS (all libraries)	17

568 GETS were required to define 17 macros.

There were no errors, warnings or information messages.

```
MACRO/LIS=LIS$:BDYSCN/OBJ=OBJ$:BDYSCN MSRC$:BDYSCN/UPDATE=(ENH$:BDYSCN)+LIB$:MACRO/LIB
```

ACTPRI LIS

ARGSCN LIS

BDYSCN LIS

CRFSUB LIS

ACTOPC LIS

ACTSTA LIS

APSECT LIS

CRFDAT LIS

ACTREF LIS

COMPUT LIS